Indian Journal of Engineering

ANALYSIS

ISSN 2319 – 7757 EISSN 2319 –7765

Awarding explore of AODV, DSDV and DSR based on VANET using sumo tool

Publication History
Received: 04 August 2015
Accepted: 21 August 2015
Published: 1 September 2015 (Special issue)

Citation
Girijalakshmi S, Sivakumar K, Chandrasekar C. Awarding explore of AODV, DSDV and DSR based on VANET using sumo tool. *Indian Journal of Engineering*, 2015, 12(30), 79-88

AWARDING EXPLORE OF AODV, DSDV AND DSR BASED ON VANET USING SUMO TOOL

S.GIRIJALAKSHMI
M.Phil
Dept. of Computer Science
Periyar University
Girijalakshmi2012@gmail.com

K.SIVAKUMAR
Ph.D Scholar
Dept. of Computer Science
Periyar University
Siva.mphil.1990@gmail.com

DR. C. CHANDRASEKAR
Professor
Dept. of Computer Science
Periyar University
ccsekar@gmail.com

Abstract

Recently many approaches have been implemented in VANET, but still there is scope of innovation. Vehicular ad-hoc network (VANET) is self-organizing helmed networks that facilitate communication in between high speed moving road vehicles like cars, trucks, busses etc. in VANET network each vehicle behave as a mobile terminal and can fold in any direction with varying speeds that makes particularly lively environment. Geographical information is used to transfer packets between intersections on the path, mitigation of path's reflection to individual node movements. The reconstruction of existing protocol or original new idea of routing in VANET environment will be a milestone and the performance evaluation will be a nice approach towards that. In this paper we evaluate performance of AODV, DSDV and DSR routing protocol on the basis performance metric packet delivery ratio, end to end delay, and throughput of the network. The Simulation parameters and routes are defined using Sumo Tool.

Keywords: VANET, SUMO TOOL, AODV, DSDV, DSR, End-to-end Delay, Throughput.

Ad_hoc On_Demand Distance Vector Routing(AODV)

Our basic proposal can be called a pure on demand route acquisition system. Nodes that do not lie on active paths neither maintain any routing information nor participate in any periodic routing table exchanges. Further, a node does not have to discover and maintain a route to another node until the two need to communicate, unless the former node is offering its services as an intermediate forwarding station to maintain connectivity between two other nodes.

When the local connectivity of the mobile node is of interest, each mobile node can become aware of the other nodes in its neighborhood by the use of several techniques, including local (not system-wide) broadcasts known as hello messages. The routing tables of the nodes within the neighborhood are organized to optimize response time to local movements and provide quick response time for requests for establishment of new routes. The algorithm's primary objectives are:

- To broadcast discovery packets only when necessary
- To distinguish between local connectivity management (neighborhood detection) and general topology maintenance
- To disseminate information about changes in local connectivity to those neighboring mobile nodes that are likely to need the information.

AODV uses a broadcast route discovery mechanism[1], as is also used (with modifications) in the Dynamic Source Routing (DSR) algorithm[2].

Instead of source routing, however, AODV relies on dynamically establishing route table entries at intermediate nodes. This difference pays off in networks with many nodes where a larger overhead is incurred by carrying source routes in each data packet. To maintain the most recent routing information between nodes, we borrow the concept of destination sequence numbers from DSDV [3]. Unlike in DSDV, however, each ad-hoc node maintains a monotonically increasing sequence number counter which is used to supersede stale cached routes. The combination of these techniques yields an algorithm that uses bandwidth efficiently by minimizing the network load for control and data traffic), is responsive to changes in topology, and ensures loop free routing.

Destination-Sequenced Distance Vector (DSDV) Protocol

Our proposed routing method allows a collection of mobile computers, which may not be close to any base station and can exchange data along changing and arbitrary paths of interconnection, to afford all computers among their number a (possibly multi-hop) path along which data can be exchanged. In addition, our solution must remain compatible with operation in cases where a base station is available. By the methods outlined in this paper, not only will routing be seen to solve the problems associated with ad-hoc networks, but in addition we will describe ways to perform such routing functions at Layer 2, which traditionally has not been utilized as a protocol level for routing, Packets are transmitted between the stations of the network by using routing tables which are stored at each station of the network. Each routing table, at each of the stations, lists all available destinations, and the number of hops to each. Each route table entry is tagged with a sequence number which is originated by the destination station.

To maintain the consistency of routing tables in a dynamically varying topology, each station periodically transmits updates, and transmits updates immediately when significant new information is available.

Routing information is advertised by broadcasting or multicasting the packets which are transmitted periodically and incrementally as topological changes are detected – for instance, when stations move within the network. Data is also kept about the length of time between arrival of the first and the arrival of the best route for each particular destination. Based on this data, a decision may be made to delay advertising routes which are about to change soon, thus damping fluctuations of the route tables. The advertisement of rout8es which may not have stabilized yet is delayed in order to reduce the number of rebroadcasts of possible route entries that normally arrive with the same sequence number.

The DSDV protocol requires each mobile station to advertise, to each of its current neighbors, its own routing table (for instance, by broadcasting its entries). The entries in this list may change fairly dynamically over time, so the advertisement must be made often enough to ensure that every mobile computer can almost always locate every other mobile computer of the collection[4].

All the computers interoperating to create data paths between themselves broadcast the necessary data periodically, say once every few seconds. In a wireless medium, it is important to keep in mind that broadcasts are limited in range by the physical characteristics of the medium. This is different than the situation with wired media, which usually have a much more well-defined range of reception.

The data broadcast by each mobile computer will contain its new sequence number and the following information for each new route:

- The destination's address;
- The number of hops required to reach the destination; and
- The sequence number of the information received regarding that destination, as originally stamped by the destination;

The transmitted routing tables will also contain the hardware address, and (if appropriate) the network address, of the mobile computer transmitting them, within the headers of the packet. The routing table will also include a sequence number created by the transmitter. Routes with more recent sequence numbers are always preferred as the basis for making forwarding decisions, but not necessarily advertised. Of the paths with the same sequence number, those with the smallest metric will be used. By the natural way in which the routing tables are propagated, the sequence number is sent to all mobile computers which may each decide to maintain a routing entry for that originating mobile computer.

One of the most important parameters to be chosen is the time between broadcasting the routing information packets. However, when any new or substantially modified route information is received by a Mobile Host, the new information will be retransmitted soon (subject to constraints imposed for damping route fluctuations), effecting the most rapid possible dissemination of routing information among all the cooperating Mobile Hosts. This quick re-broadcast introduces a new requirement for our protocols to converge as soon as possible. It would be calamitous if the movement of a Mobile Host caused a storm of broadcasts, degrading the availability of the wireless medium.

Mobile Hosts cause broken links as they move from place to place, The broken link may be detected by the layer-2 protocol, or it may instead be inferred if no broadcasts have been received for a while from a former neighbor. A broken link is described by a metric of ∞ (i.e., any value greater than the maximum allowed metric). When a link to a next hop has broken, any route through that next hop is immediately assigned an ∞ metric and assigned an updated sequence number. Since this qualifies as a substantial route change, such modified routes are immediately disclosed in a broadcast routing information packet.

Building information to describe broken links is the only situation when the sequence number is generated by any Mobile Host other than the destination Mobile Host. Sequence numbers defined by the originating Mobile Hosts are defined to be even numbers, and sequence numbers generated to indicate cm metrics are odd numbers. In this way any “real” sequence numbers will supersede an ∞ metric.

When a node receives an co metric, and it has a later sequence number with a finite metric, it triggers a route update broadcast to disseminate the important news about that destination. In a very large population of Mobile Hosts, adjustments will likely be made in the time between broadcasts of the routing information packets. In order to reduce the amount of information carried in these packets, two types will be defined. One will carry all the available routing information, called a “full dump”. The other type will carry only information changed since the last full dump, called an “incremental”. By design, an incremental routing update should fit in one network protocol data unit (NPDU). The full dump will most likely require multiple NPDUS, even for relatively small populations of Mobile Hosts.

Full dumps can be transmitted relatively infrequently when no movement of Mobile Hosts is occurring, When movement becomes frequent, and the size of an incremental approaches the size of a NPDU, then a full dump can be scheduled (so that the next incremental will be smaller). It is expected that mobile nodes will implement some means for determining which route changes are significant enough to be sent out with each incremental advertisement. For instance, when a stabilized route shows a different metric for some destination, that would likely constitute a significant change that needed to be advertised after stabilization.

One solution is to delay the advertisement of such routes, when a Mobile Host can determine that a route with a better metric is likely to show up soon. The route with the later sequence number must be available for use, but it does not have to be advertised immediately unless it is a route to a destination which was previously unreachable. Thus, there will be two routing tables kept at each Mobile Host; one for use with forwarding packets, and another to be advertised via incremental routing information packets. To determine the probability of imminent arrival of routing information showing a better metric, the Mobile Host has to keep a history of the weighted average time that routes to a particular destination fluctuate until the route with the best metric is received. We hope that such a procedure will allow us to predict how long to wait before advertising new routes.

DSR Protocol Description Overview and Important Properties of the Protocol

The DSR protocol is composed of two mechanisms that work together to allow the discovery and maintenance of source routes in the ad hoc network:

- *Route Discovery* is the mechanism by which a node **S** wishing to send a packet to a destination node **D** obtains a source route to **D**. Route Discovery is used only when **S** attempts to send a packet to **D** and does not already know a route to **D**.
- *Route Maintenance* is the mechanism by which node **S** is able to detect, while using a source route to **D**, if the network topology has changed such that it can no longer use its route to **D** because a link along the route no longer works. When Route Maintenance indicates a source route is broken, **S** can attempt to use any other route it happens to know to **D**, or can invoke Route Discovery again to find a new route. Route Maintenance is used only when **S** is actually sending packets to **D**.

Route Discovery and Route Maintenance each operate entirely *on demand*. In articular, unlike other protocols, DSR requires *no* periodic packets of *any kind* at *any level* within the network. For example, DSR does not use any periodic routing advertisement, link status sensing, or neighbor detection packets, and does not rely on these functions from any underlying protocols in the network. This entirely on-demand behavior and lack of periodic activity allows the number of overhead packets caused by DSR to scale all the way down to *zero*, when all nodes are approximately stationary with respect to each other and all routes needed for current communication have already been discovered. As nodes begin to move more or as communication patterns change, the routing packet overhead of DSR *automatically* scales to only that needed to track the routes currently in use.

In response to a single Route Discovery (as well as through routing information from other packets overheard), a node may learn and cache multiple routes to any destination. This allows the reaction to routing changes to be much more rapid, since a node with multiple routes to a destination can try another cached route if the one it has been using should fail. This caching of multiple routes also avoids the overhead of needing to perform a new Route Discovery each time a route in use breaks.

The operation of Route Discovery and Route Maintenance in DSR are designed to allow uni-directional links and asymmetric routes to be easily supported. DSR allows such uni-directional links to be used when necessary, improving overall performance and network connectivity in the system. DSR also supports internetworking between different types of wireless networks, allowing a source route to be composed of hops over a combination of any types of networks available [5]. For example, some nodes in the ad hoc network may have only short-range radios, while other nodes have both short-range and long-range radios.

The combination of these nodes together can be considered by DSR as a single ad hoc network. In addition, the routing of DSR has been integrated into standard Internet routing, where a “gateway” node connected to the Internet also participates in the ad hoc network routing protocols; and has been integrated into Mobile IP routing, where such a gateway node also serves the role of a Mobile IP foreign agent [6, 7].

Basic DSR Route Discovery

When some node **S** originates a new packet destined to some other node **D**, it places in the header of the packet a *source route* giving the sequence of hops that the packet should follow on its way to **D**. Normally, **S** will obtain a suitable source route by searching its *Route Cache* of routes previously learned, but if no route is found in its cache, it will initiate the Route Discovery protocol to dynamically find a new route to **D**. In this case, we call **S** the *initiator* and **D** the *target* of the Route Discovery.

For example, Figure 1 illustrates an example Route Discovery, in which a node **A** is attempting to discover a route to node **E**. To initiate the Route Discovery, **A** transmits a ROUTE REQUEST message as a single local broadcast packet, which is received by (approximately) all nodes currently within wireless transmission range of **A**. Each ROUTE REQUEST message identifies the initiator and target of the Route Discovery, and also contains a unique *request id*, determined by the initiator of the REQUEST.

Each ROUTE REQUEST also contains a record listing the address of each intermediate node through which this particular copy of the ROUTE REQUEST message has been forwarded. This route record is initialized to an empty list by the initiator of the Route Discovery.

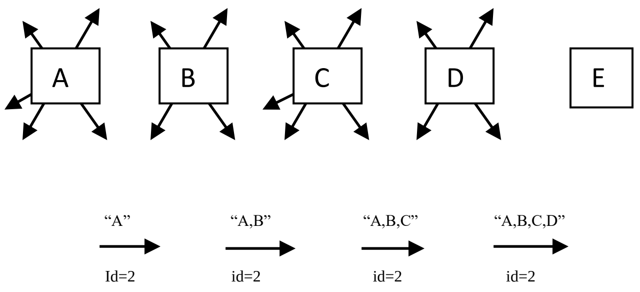

Figure 1 - **Route Discovery example: Node A is the initiator, and node E is the target.**

When another node receives a ROUTE REQUEST, if it is the target of the Route Discovery, it returns a ROUTE REPLY message to the initiator of the Route Discovery, giving a copy of the accumulated route record from the ROUTE REQUEST; when the initiator receives this ROUTE REPLY, it caches this route in its Route Cache for use in sending subsequent packets to this destination. Otherwise, if this node receiving the ROUTE REQUEST has recently seen another ROUTE REQUEST message from this initiator bearing this same request id, or if it finds that its own address is already listed in the route record in the ROUTE REQUEST message, it discards the REQUEST.

In returning the ROUTE REPLY to the initiator of the Route Discovery, such as node **E** replying back to **A** in Figure 1, node **E** will typically examine its own Route Cache for a route back to **A**, and if found, will use it for the source route for delivery of the packet containing the ROUTE REPLY. Otherwise, **E** may perform its own Route Discovery for target node **A**, but to avoid possible infinite recursion of Route Discoveries, it must piggyback this ROUTE REPLY on its own ROUTE REQUEST message for **A**. It is also possible to piggyback other small data packets, such as a TCP SYN packet [8], on a ROUTE REQUEST using this same mechanism.

Node **E** could also simply reverse the sequence of hops in the route record that it trying to send in the ROUTE REPLY, and use this as the source route on the packet carrying the ROUTE REPLY itself. For MAC protocols such as IEEE 802.11 that require a bi-directional frame exchange as part of the MAC protocol [9].

This route reversal is preferred as it avoids the overhead of a possible second Route Discovery, and it tests the discovered route to ensure it is bi-directional before the Route Discovery initiator begins using the route. However, this technique will prevent the discovery of routes using uni-directional links. In wireless environments where the use of uni-directional links is permitted, such routes may in some cases be more efficient than those with only bi-directional links, or they may be the only way to achieve connectivity to the target node.

When initiating a Route Discovery, the sending node saves a copy of the original packet in a local buffer called the *Send Buffer*. The Send Buffer contains a copy of each packet that cannot be transmitted by this node because it does not yet have a source route to the packet's destination. Each packet in the Send Buffer is stamped with the time that it was placed into the Buffer and is discarded after residing in the Send Buffer for some timeout period; if necessary for preventing the Send Buffer from overflowing, a FIFO or other replacement strategy can also be used to evict packets before they expire.

Sumo Tool

The development of "**S**imulation of **U**rban **Mo**bility", or "SUMO" for short, started in the year 2000. The major reason for the development of an open source, microscopic road traffic simulation was to support the traffic research community with a tool into which own algorithms can be implemented and evaluated with, without the need to regard all the artifacts needed to obtain a complete traffic simulation, such as implementing and/or setting up methods for dealing with road networks, demand, and traffic controls. By supplying such a tool, the DLR wanted to i) make the implemented algorithms more comparable, as a common architecture and model base is used, and ii) gain additional help from other contributors.

Since 2001, with the first running version, SUMO has been used within a large number of projects done within the DLR. The main application was to implement and evaluate traffic management methods, such as new traffic light systems or new traffic guidance approaches. Additionally, SUMO was used for short-term (30min) traffic forecast during large events with many participants, and was used for evaluating traffic surveillance using GSM networks.

Since 2002, SUMO is also in use at other institutions. Here, the major interest seems to be the evaluation of vehicle-to-vehicle and vehicle-to-infrastructure communication. Two major third-party projects should be mentioned in this context, the first, TraCI, is an extension of SUMO by the possibility to communicate with external applications, done at the University of LÃ¼beck by Axel Wegener. The second project with a high impact is "TraNS", a direct coupling between SUMO and the network simulator ns2 which uses TraCI for communication and that was set up by Michal Piorkowski and Maxim Raya at the EPFL Lausanne[10].

SIMULATION WORK

Simulation Parameters

Parameter Type	**Value**
Network Simulator	NS-2.35
Routing Protocol	DSDV, AODV, DSR
Simulation Time	50ms, 100ms, 500ms
Simulation Area	1000 * 1000 m
Number of Nodes	10
DATA TYPE	FTP
Packets Generation Rate	80 kb
Packet Size	160 bytes
MAC Protocol	IEEE802.11
Channel Type	Wireless Channel

SUMO TOOL RUNNING

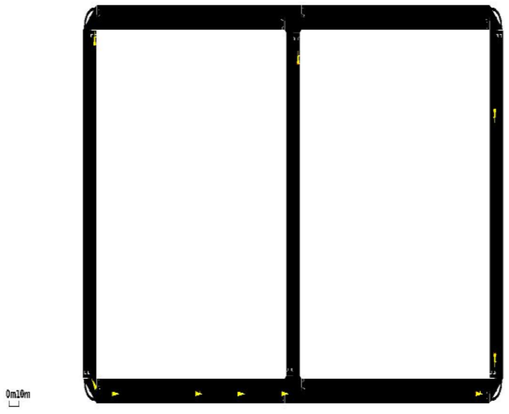

Figure 2- SUMO TOOL Running window

FORMULA

- Packet Delivery Ratio:

$$PDR = \frac{No.of\ packet\ Recieve}{No.of\ Packet\ Send}\ X\ 100$$

- End to End Delay:

EED = Receive Packet Time - Send Packet time

- Throughput:

$$TP = \left(\frac{Receive\ Size}{Stop\ Time - Start\ time}\right) X\ 8/1000$$

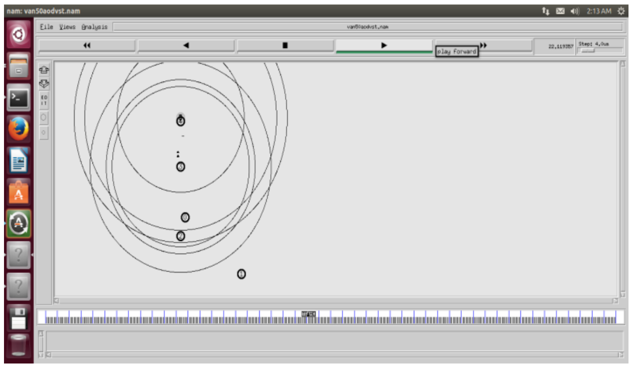

Figure 3 – AODV NAM - 50

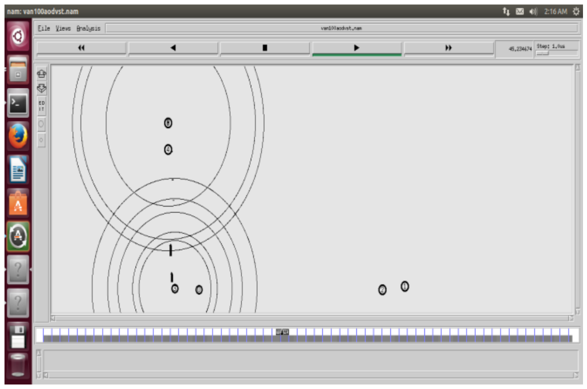

Figure 4 – AODV NAM-100

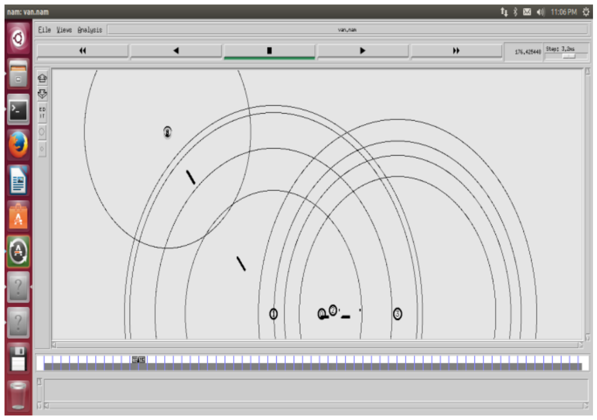

Figure 5 – AODV NAM - 500

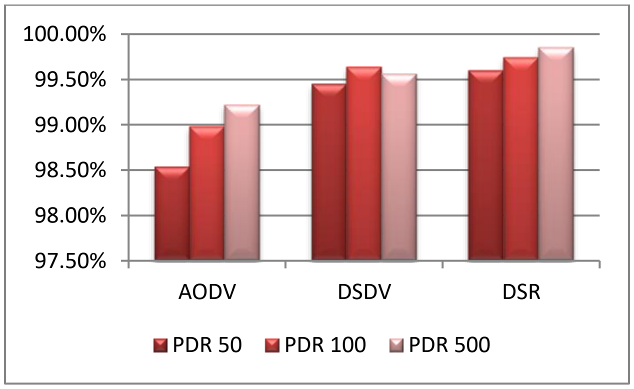

Chart 1 - PDR

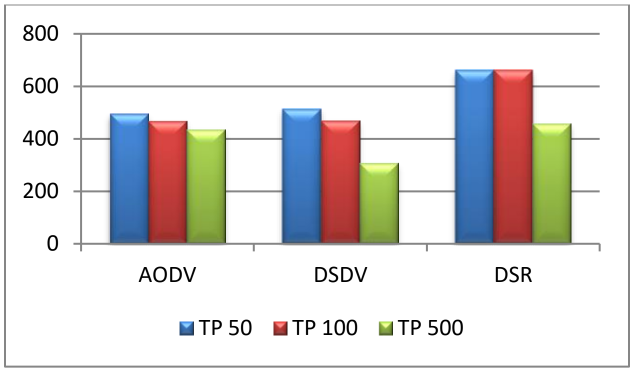

Chart 2 - THROUGHPUT

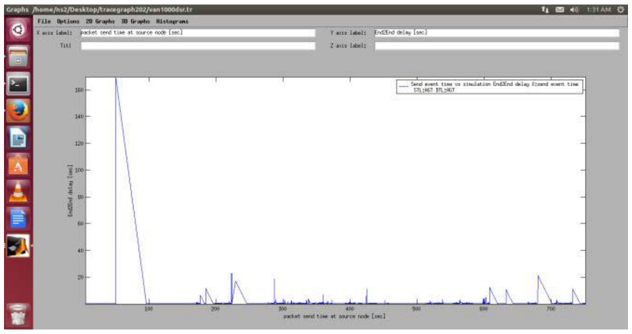
Figure 9 – END TO END DELAY FOR DSR-500

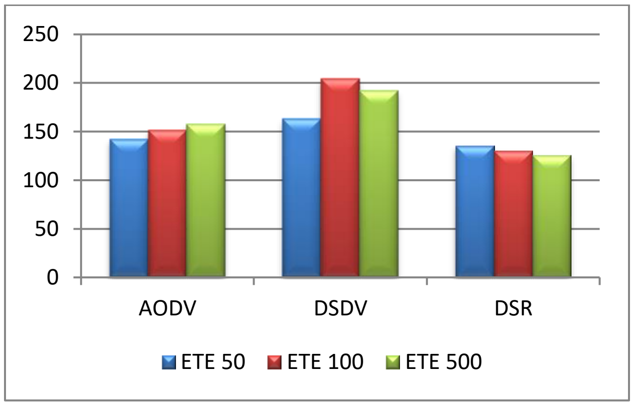

Chart 3 – END TO END DELAY

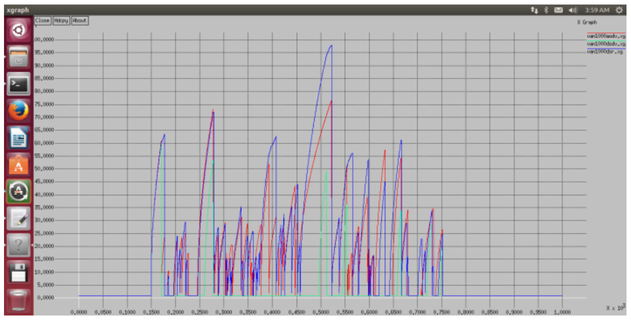
Figure 6 – AODV,DSDV & DSR COMPARISON OF THROUGHPUT

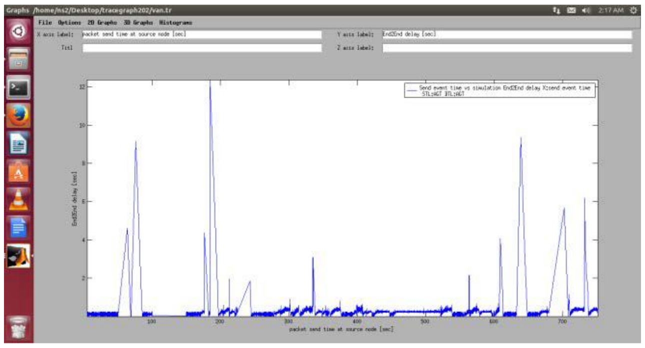
Figure 7 – END TO END DELAY FOR AODV-500

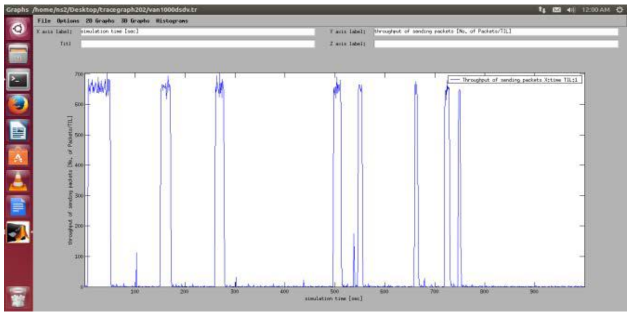
Figure 8 – END TO END DELAY FOR DSDV-500

CONCLUTION

In this study, compare to AODV & DSDV protocols DSR reflects High Throughput and Pocket Delivery Ratio. Meanwhile AODV & DSDV protocols DSR Reflects low in End to End delay. In future we enhance our criterion level for comparison. We also evaluate some other protocols with DSR, DSDV and AODV protocols.

References:

1. M.S. Corson and A. Ephremides, A Distributed Routing Algorithm for Mobile Wireless Networks ACMJ Wireless Networks(1), Jan 1995.
2. D. Johnson and D. Maltz Dynamic source routing in ad-hoc wireless networks In Computer Communications Review- Proceedings of SIGCOMM 96,Aug 1996.

3. S. Guha and S. Khuller, Approximation Algorithms for Connected Dominating Sets. University of Maryland College Park Technical Report 3660, June 1996.
4. J.J. Garcia Luna-Aceves, A Unified approach to loop-free routing using distance vectors or link states. In ACM SIGCOMM, pages 212-223,1989.
5. Josh Broch, David A. Maltz, and David B. Johnson. Supporting Hierarchy and Heterogeneous Interfaces in Multi-Hop Wireless Ad Hoc Networks. In *Proceedings of The International Symposium on Parallel Architectures, Algorithms and Networks* (ISPAN'99), Workshop on Mobile Computing, Perth, Western Australia, June 1999. IEEE Computer Society
6. David B. Johnson. Scalable Support for Transparent Mobile Host Internetworking. *Wireless Networks*, 1(3):311–321, October 1995.
7. Charles Perkins, editor. IP Mobility Support. RFC 2002, October 1996.
8. J. B. Postel, editor. Transmission Control Protocol. RFC 793, September 1981.
9. IEEE Computer Society LAN MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std 802.11-1997. The Institute of Electrical and Electronics Engineers, New York,1997.
10. SourceForge.net: SUMO User Documentation – sumo